Untraceability in Mobile Networks *

Didier Samfat Refik Molva
EURECOM Institute
2229, Route des Crêtes
06904 Sophia-Antipolis - FRANCE
{*molva,samfat*}*@eurecom.fr*

N. Asokan
Department of Computer Science
University of Waterloo
Waterloo, Ont. N2L 3G1, CANADA
nasokan@uwaterloo.ca

Abstract - **User mobility is a feature that raises many new security-related issues and concerns. One of them is the disclosure of a mobile user's real identity during the authentication process, or other procedures specific to mobile networks. Such disclosure allows an unauthorized third-party to track the mobile user's movements and current whereabouts. Depending on the context, access to any information related to a mobile user's location without his consent can be a serious violation of his privacy. This new issue might be seen as a conflicting requirement with respect to authentication: untraceability requires hiding the user's identity while authentication requires the user's identity to be revealed in order to be proved. What is needed is a single mechanism reconciling both authentication and privacy of a mobile user's identification. The basic solution to this problem is the use of *aliases*. Aliases insure untraceability by hiding the user's real identity as well as his relationship with domain authorities. In this paper, we present a classification scheme to identify the various degrees of untraceability requirements. We then present an efficient method for the computation of aliases and apply it to a new set of inter-domain authentication protocols. We demonstrate that these protocols can be designed to meet various degrees of untraceability requirements. In designing these protocols, we try to avoid the drawbacks of authentication protocols in existing mobile network architectures such as CDPD and GSM.**

Keywords: authentication, anonymity, mobility, security, GSM, CDPD, alias, location privacy

***The work describes herein was funded by IBM Zürich Research Laboratory**

1 Introduction

Digital information is becoming more and more important in everyday life. People are often asked to provide identification information about themselves to organizations in order to obtain some service. Many examples can be found in existing systems: payment for goods and services with credit cards, access to the Internet etc. A common factor in these environments is that during any such transaction, the user has to provide a claimed identity to the system. If no care is taken, an eavesdropper may be able to watch the message exchanges (on the air interface in a cellular environment or through the signalling protocols exchanged on the wired network) and thus infer information such as who is involved in a specific transaction, where and when it was performed etc.

In mobile networks, a similar problem that arises due to mobility is the unauthorized tracking of users' migration. Regardless of the type of network (i.e. wireless/cellular networks and wired networks supporting mobility), a typical situation arises when a mobile user (with or without a device) registered in a *home domain*[1] appears ("pops up") in a new foreign domain. In order to obtain a service, the mobile user may need to prove his good standing to the visited domain. A common solution is to require that the user authenticate himself to the home domain which then confirms the solvency of the user in the visited domain. Usually, during this process the user has to provide a non-ambiguous identity to the home domain and prove it.

An intuitive solution is to assign *aliases* to the mobile user. However, hiding only the user's real identity is not sufficient to fulfill all untraceability requirements. The user's real identity may be inferred from the user's relationship with domain authorities based on mobility management messages. Therefore, providing untraceability to a user involves protecting information related to his identity, location, and movements. The degree of untraceability required depends on various factors such as the security policy being enforced, the cost vs. benefit tradeoffs etc. For complete untraceability, no entity other than the user should know any information regarding the user's identity, location or movements. Depending on the circumstances, a lesser degree of untraceability may be satisfactory.

In this paper, first we present existing mechanisms for providing untraceability in banking systems, fol-

[1] A *domain* consists of regions and entities that fall within a common administrative control.

lowed by a classification of the different levels of untraceability requirements. We then discuss other issues in mobile networks, such as user interface and resource limitations, that have an impact on the provision of untraceability. Finally, we present a summary of the drawbacks of solutions in existing mobile networks and then describe and evaluate a set of strong untraceable authentication protocols suitable for mobile users.

2 Anonymity in Other Contexts

Provision of anonymity is not a new feature. David Chaum [10, 11] and others have done extensive work in developing techniques for secure, untraceable electronic transactions in banking environments or in fixed networks. In this section, we describe some of Chaum's approaches in particular and discuss their applicability to mobile computing environments. The following is a bare-bones description of the detailed techniques outlined in [11].

The first mechanism provides untraceability of payment transactions. A bank announces public keys corresponding to various denominations of money. A customer can convert his money into a "digital coin" by generating a serial number (which is derived from a random number), "blinding" it with another random number and getting the bank to sign the blinded quantity with the public key of the desired denomination. The bank will deduct the amount of money corresponding to the chosen denomination. The customer then "unblinds" the signed digital coin by removing the blinding random number. Now, the digital coin can be given to a shop along with the corresponding serial number. The shop can verify that it is indeed a valid coin since it knows the bank's public key. When the shop presents the digital coin to the bank, the bank will credit the shop's account and record the serial number of the coin in order to disallow double spending. The bank cannot link the coin and the customer because it cannot know the serial number when it signed the coin.

The approach is essentially a capability-based approach which obviates the need for authentication. The second approach allows a user to obtain credentials from an organization and present them to a *different* organization without either organization being able to link these activities to the same user. The user identifies himself by a different alias with each different organization. He can obtain a credential Cr_A from Organization A, to which he is known by the alias a. The credential Cr_A is essentially a signature using A's secret key. However, it also contains information about the various aliases by which the user is known to different organizations. Organization A, however, cannot extract these other aliases during the signing process. This is achieved by using the same blind signature technique described earlier. The user can present Cr_A to a different organization B to which it is known by the alias b. B will be able to verify that Cr_A was in fact signed by A and that it belongs to b. But it cannot infer any information about its other aliases.

One could envision ways in which these techniques can be adapted to a mobile network. The primary problem with these techniques is the amount of resources they require. Storage and processing capacity is typically at a premium on mobile devices. In the case of wireless networks, network bandwidth is also a limited resource. Secondly, these techniques strive to provide *complete* untraceability. This may not be acceptable under all security policies. For example, complete untraceability also sets the stage for "perfect crimes." This calls for a general approach that can provide different levels of untraceability without imposing extensive resource requirements that might render the solution impractical in mobile networks.

3 Classification of Untraceability Requirements

Anonymity can be defined according to two different dimensional parameters: information related to the identification of the user and entities which are able to have access to these pieces of information.

As we alluded to in the preceding sections, the required level of untraceability depends on various factors like the cost incurred by providing this service, the perceived benefits from such a service, practical constraints and so on. To help choose the level of untraceability necessary for a given environment, it is necessary to develop a classification scheme to represent the various possible levels of untraceability requirements.

A specific untraceability requirement is represented in terms of a two dimensional matrix. If a particular class of entities knows (or can infer during the course of a protocol run based solely on the messages exchanged during runs of the protocol) a particular piece of information, the corresponding table entry is marked 1. Otherwise, it is marked 0.

In the case of mobile networks, the different classes of entities which might know identification information are: the user (U), the home domain (H), the remote domain (R), legitimate network entities (L) (such as other authorized third parties involved in a transaction) and eavesdroppers (E) (unauthorised third parties). Since U is a trivial class containing just one entity that *always* has access to all the traceability information about itself, we omit it in subsequent discussion.

The above entities may have access to one or more of the following pieces of information: the full identity of the user f, the identity of the home domain (affiliation) h, and the identity of the remote domain r.

This scheme gives rise to a whole spectrum of untraceability levels. We identity five particular cases of interest as to the knowledge of relationship between the entities and the identification information.

- C_1: **Hiding User Identity from Eavesdroppers.** Most of the existing solutions address this requirement. The resulting policy can be formulated as follows:

	H	**R**	**L**	**E**
f	*1*	*1*	*1*	**0**
h	*1*	*1*	*1*	*1*
r	*1*	*1*	*1*	*1*

  In the Global System for Mobile communications(GSM) [1], the use of Temporary Mobile System Identifiers (TMSIs) is intended to meet this requirement. When the user appears for

the very first time in a new foreign domain, she needs to establish temporary residency with the foreign administrative authority. In this step, a long-term alias can be assigned to the user for the duration of his stay. The main problem with this approach is that all activity performed in the remote domain can be linked to this single alias. After a while, the relationship between this alias and the user's home domain may be discovered by traffic analysis.

However, a basic requirement is that derivation of successive aliases should not lead to the disclosure of the real identity. We need to envision different ways of assigning aliases to mobile users during their migration so that this additional requirement is met as well. A more secure alternative is to assign a different alias each time the user accesses a service in the visited domain. This avoids the disclosure of the user's relationship with the foreign authority.

- C_2: **Hiding User Identity from Foreign Authorities.** In some situations, there is no need for the foreign authority to know the real identity of the user – it may only need proof of the solvency of the user accessing the service and enough information to bill the user's home authority. In this case, the policy is:

	H	R	L	E
f	*1*	**0**	**0**	**0**
h	*1*	*1*	*1*	*1*
r	*1*	*1*	*1*	*1*

- C_3: **Hiding Relationship Between the User and Authorities.** In a higher level of privacy, it is important to protect the existing relationship between the mobile user and his home authority from a third-party. This policy is:

	H	R	L	E
f	*1*	**0**	**0**	**0**
h	*1*	*1*	**0**	**0**
r	*1*	*1*	*1*	*1*

The real identity of the user may be discovered by analysing the traffic between the foreign and the home authorities. In other words, each time the user accesses the network, if the identify of his home authority is not protected, information about the user's real identity may be inferred.

For instance, if an aliased user x visiting a remote domain in France wants to authenticate to his home domain WhiteHouse.Gov and an intruder happens to know that the only users from WhiteHouse.Gov currently in France are President@WhiteHouse.Gov and Vice.President@WhiteHouse.Gov, the intruder can conclude that x in fact corresponds to one of these two real identities.

- C_4: **Hiding the Identity of the Home Authority from Foreign Authorities.** When the mobile user needs to be authenticated in a foreign domain, the foreign authority needs to contact the user's home authority in order to confirm the good standing of the user. Therefore, even if the real identity of the user is hidden, his relationship with his home is known by the foreign authority. However, in environments where there are other means of establishing solvency (or where solvency is not an issue), a higher level untraceability is possible by preventing even the foreign authority from knowing the identity of the home authority. This policy corresponds to:

	H	R	L	E
f	*1*	**0**	**0**	**0**
h	*1*	**0**	**0**	**0**
r	*1*	*1*	*1*	*1*

- C_5: **Hiding User Behavior from Home Authority.** In some cases it might be important for a mobile user to hide his migration from his home authority. This requirement is especially important if perfect secrecy of user behavior should be guaranteed by the system, that is, if no one other than the user should know about the user's location. The resulting policy can be formulated as follow:

	H	R	L	E
f	**0**	**0**	**0**	**0**
h	*1*	**0**	**0**	**0**
r	**0**	*1*	*1*	*1*

In other words, no entity has any information about the user. This principle of course would contrast the intent of a "big brother" towards global observation of users' behavior.

It can be noticed that the requirements defined by each class C_i form a subset of class C_{i+1} because class C_{i+1} is obtained by increasing the constraints of class C_i.

Note that classes C_4 and C_5 may be contradictory with other (unrelated to security) system requirements. For example, when the user needs to be reachable at any moment by a single identification[2], the home domain always needs to know the location of the mobile user in order to route the incoming calls towards the user. In this situation, classes C_4 and C_5 can be conflicting requirements with the mandatory need of the network to track the mobile user.

Another situation arises when the home domain is the entity that is expected to vouch for the solvency of a user while he is traveling. In this case, the home domain must always have the possibility to revoke the mobile user's account at any time. However, it is not always the case that solvency needs to be underwritten by the home domain. For example, if the original solvency guarantee from the home domain contains limits (in terms of both amount of resources and time) and all domain level servers trust each other to some extent, a foreign domain can rely on the solvency guarantee from the previous domain from which the user wandered in. Solvency may also be defined in

[2]for instance a phone number.

non-monetary terms using an anonymous capability scheme in order to perform access control. For example, a domain may be willing to provide free services to members of a club. In this case, a foreign domain may again trust any other domain level server to vouch for a user's membership in the club. In such situations, C_4 and C_5 requirements become meaningful.

4 Implications of Untraceability in Mobile Networks

Various factors, both technical and non-technical, influence the level of untraceability that is appropriate in a given environment. In this section, we describe how hardware limitations and organizational policy regarding service provisions influence the application of the alias solution.

4.1 Aliases and End-User Interface

In the case of a network that supports mobility of users (but not of computing devices), a user may have only a password or PIN for the purpose of authentication [9]. Users who need to provide their credentials in order to access services are forced to rely on the available public access equipment (i.e workstation or public terminal). We can reduce this exposure by using traveling aliases in order to avoid revealing the real identity of the user to entities under the control of the visited domain. Therefore, even if the public access equipment knows the password of the user, it does not know *whose* credential it is.

The alias used should be a character string as easy to remember as the usual user-name. The only condition on the generation of this alias is that it should not be related to the user-name at the home domain. The choice of such an alias is not subject at all to the same considerations as the generation of a secret password. In the case of passwords, the security requirement is to make guessing the value hard. The goal is to choose an unusual string as a password so that a straightforward search through the list of known words would not yield its value. As aliases are sent in clear text, there is no need to use unusual strings; common words of the dictionary are sufficient.

Nevertheless, a similar attack can be made in this kind of situation. Having the password of the user, a malicious workstation can scan a list of pre-established aliases in order to know whom the password belongs to. Little can be done in this situation unless the user changes his alias regularly. In that case, the user will need a list of traveling aliases that are easy to remember.

In networks that support portable computing devices, a mobile user in possession of a trusted device (e.g., smartcard, portable phone) can benefit from reduced exposure by having better random aliases which can be changed more frequently and in a transparent way. In fact, the end-user terminal can share additional specific information with his home authority in order to generate strong random aliases assuming that the mobile unit has non volatile memory.

4.2 Aliases vs. Accounting and Billing

Provision of higher levels of untraceability requires that the foreign domain authority be kept "in the dark" about the identities of visiting users. This level of untraceability can be undesirable when accounting and billing are involved. In the case of classes C_1, C_2 and C_3, the foreign authority can still keep a trace of the user by recording the proof of use and later asking the home authority for a "refund."

However, classes C_4 and C_5 are conflicting requirements with the need for billing. First, if the foreign authority does not know the identity of the home domain, it will not be able to later bill the mobile user. Note that class C_5 also conflicts with the need for the home to forbid a user to consume resources in a foreign domain, after having revoked the account privileges of the user.

Alias solutions are not in contradiction with accountability/billing provided that they allow the home authority to recover the real user's identity in order to invoice him or by having recourse to sophisticated techniques based on digital cash as described in [10].

5 Review of Existing Approaches

In this section, we review existing approaches for untraceability in mobile networks.

The Global System for Mobile (GSM) [1] is the first digital cellular network to provide anonymity to its subscribers. In GSM, untraceability is provided by using aliases known as Temporary Mobile Subscriber Identifiers (TMSI). The main concern with GSM is when a user first switches on his portable phone: his real identity, known as the International Mobile Subscriber Identifier (IMSI), is transmitted in the clear through the radio path (the TMSI is only allocated after this step). In this case, if the user is continuously tracked, his real identity is revealed; it is therefore possible for an eavesdropper to correlate this IMSI with the TMSIs assigned subsequently. A similar situation arises when synchronization of TMSI between the user and the home entity is lost – the user is again forced to send his IMSI in the clear.

Another point of contention with GSM is that the fixed network is assumed to be secure, and all visited domains know the real identity of the mobile unit. Location data are transmitted in the clear through the fixed network, thereby depriving the user of identity privacy.

In contrast to GSM, Cellular Digital Packet Data (CDPD) [8] has a more secure approach. Before the authentication procedure takes place, the mobile unit engages a Diffie-Hellman key exchange protocol in order to share a secret session key with the foreign authority. Next, the mobile unit enciphers its identity with this new key and transmits it to the foreign authority, which deciphers the encrypted identity with the same shared secret key.

The first drawback of this approach is that it allows the foreign authority to know the real identity of the mobile unit. The second drawback remains the nature of the Diffie-Hellman protocol, which allows an intruder to masquerade as the foreign authority (using what is called the "man-in-the-middle" attack) and to discover the mobile unit's real identity, among other things.

With respect to the requirement classification of Section 3, both GSM and CDPD only partially covers the requirements of case C_1. Even if these approaches are reasonable in their limited contexts, they are not sufficient if higher levels of untraceability is required. Providing total anonymity to mobile users requires hiding the user's real identity from both unauthorized parties (eavesdroppers) and authorized parties (remote administrative authorities) as described

in Section 3.

6 Reconciling Authentication and Untraceability

Untraceability might be seen as a conflicting requirement with respect to authentication, as untraceability requires hiding the user's identity in contrast with authentication that requires the user's identity to be revealed in order to prove it. In this section we present a solution for the computation of aliases which can be used to protect the identities of the different parties involved in the authentication process.

6.1 Initial Assumptions

We begin by stating that a user has one *home* which is the administrative domain where he is registered on a long-term basis. Moreover, when accessing the network in each visited domain, the mobile user is authenticated with a traditional server-based authentication mechanism such as Kerberos [2] or KryptoKnight [3]. Users of a given network domain are registered with that domain's Authentication Server (AS). The AS of a domain can be replicated or partitioned within the domain but the set of all partitioned and duplicated ASs represents a single domain-level authority.

We assume that the user has a personal device which can store information in a non-volatile memory, because little can be done for untraceability of mobile users having only their user-name and password (or PIN) for authentication as described in Section 4.

Moreover, the user needs a universal identification (for example home user identification) to which only the home domain can link the different aliases. This identification can be a number or a string allocated to the user at subscription time. This is particularly important for a central authority, especially when accounting and billing are involved.

6.2 Design Criteria

In order to insure good anonymity to the mobile user during his migration, the alias generation must take into account the following design criteria:

- *One-time-use alias.* Long-term use of a single static alias is not a good solution, as it may be correlated to the user's real identity. Consequently, it is desirable to use a different alias for each security process.
- *No direct relationship between aliases.* This is quite an obvious but important requirement, as we want to hide the user identification effectively.
- *Domain separation.* Even when assuming conspiracy of all visited domains (except the home domain) the real identity of the user should not be discovered.

The solution should also allow the protection of the identity of some or all of the authorities involved in the authentication, in order to fulfill some of the untraceability requirements as described in Section 3.

6.3 Protocol Building Blocks

We base our design on the one-way authentication protocol (see Figure 1) borrowed from *KryptoKnight*, an authentication and key distribution service developed at IBM Research [3]. The reason for such a choice is that KryptoKnight benefits from having strong authentication protocols, and provides formal insurance of security with respect to a number of attacks. The cryptographic messages used in KryptoKnight present some qualities that make them attractive for use in building strong untraceable authentication protocols.

Due to space constraints, we do not go into the details of and the rationales behind the KryptoKnight approach. The reader is referred to the various KryptoKnight papers (e.g. [3]) for more information.

The cryptographic token in Figure 1 is computed by applying a strong encryption function E, e.g., DES [7], with K_{ab} as the encryption key, over three inputs: a nonce (N_a), a timestamp (T_a) and the name of the message originator (A). The $\bigoplus$ symbol indicates a bitwise exclusive-OR operation. In the rest of the paper $AUTH_{ab}$ will denote the one-way authentication message of an initiator A to a responder B:

$$AUTH_{ab} = [N_a, T_a, Token_{K_{ab}}(A, T_a, N_a)]$$

In Figure 2, the initiator A is sending a ticket to a responder B containing a session key K_s to be shared by B with a third party, C. In the following sections, $TICK_{K_{ab}}(K_s)$ will denote the expression of this certificate:

$$TICK_{K_{ab}}(A, B, C, K_s) = Token_{Kab}(N_a \bigoplus C, N_b, N_a \bigoplus A) \bigoplus K_s$$

6.4 Alias Computation

The KryptoKnight protocols [5] do not provide identity privacy as the initiator A sends his identity in the clear to the responder B. Therefore, if we want to protect an identity A from an unauthorized party, we need to compute an alias which only the responder B can understand. The basic idea of this solution uses the following fact. Use of shared secret keys for authentication requires that a claimed identity be provided whereas in public key cryptosystems the identity of the prover can be implicit.

An alias for the principal A can be computed as follows: [3]

$$P_b(N_a^{'}, N_a^{'} \bigoplus A)$$

$P_b()$ denotes the result of the encryption with the responder's public key over two inputs: one nonce and the identity of the initiator. Upon receiving the alias, B obtains $(N_a^{'}, N_a^{'} \bigoplus A)$ by deciphering the alias with his secret key S_B, then the real identity A is retrieved by computing $N'_a \bigoplus N'_a \bigoplus A$.

7 Untraceable Authentication Protocols

Many cryptographic solutions are possible for the problem of ensuring anonymity to mobile users. The main distinguishing factor is the level of untraceability needed. We develop three authentication protocols taking into account the five classes of privacy as described in Section 3. In doing so, we avoid the drawbacks of GSM and CDPD. In a mobile computing environment the proposed solutions are more suitable than the "digital cash" approach for reasons related to performance and system requirements as mentioned in Section 2.

[3] We use the $\bigoplus$ operator in order to provide additional resistance against known-plaintext attacks without making any assumptions about the cryptosystem used (e.g., the block size). In theory, the $\bigoplus$ opreation is not required; the alias can be $P_b(N_a^{'}, A)$.

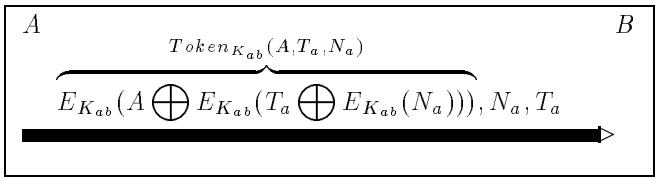

Figure 1: One-Way Authentication Protocol

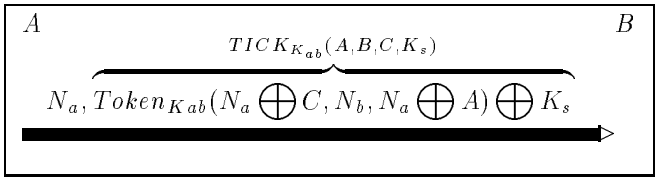

Figure 2: Ticket containing a secret shared key K_s

7.1 Basic Untraceable Protocol

The basic untraceable authentication protocol is depicted in Figure 3. The main idea is to allow the user to change his alias on successive security transactions by generating a random alias each time. The following notation is used in this protocol:

- Uid – Universal identification of the end-user U in his home domain
- Uid_x – Identification of the user in domain X
- AS_h, AS_r – Authentication Servers of the home domain and of the remote domain respectively
- K_u – Strong key shared by U and AS_h
- K_{rh} – Long-term key shared between AS_r and AS_h
- K_{ur} – Location-dependent key (result of a strong one-way-hash function $F(U, AS_r, K_u)$) to be used by U with AS_r
- P_x, S_x – Public key, Secret key pair of AS_x
- N_x – Nonce issued by entity X
- $P_x(M)$ – Encryption of message M with the public key P_x of AS_x
- $AUTH_{XY}$ – Authentication message computed by X and to be verified by Y. $AUTH_{XY}$ is a challenge message composed of a clear-text part and an authentication token. The exact format of $AUTH_{XY}$ is described in Figure 1.
- $TICK_{K_x}(K_s)$ – A ticket computed with the key K_x and containing a session key K_s
- $F(M)$ – Strong one-way hash function such as MD5 [4] applied on message M
- $\bigoplus$ – exclusive-or operation (xor).

This protocol provides class C_1 and C_2 of untraceability as the user's real identity is not revealed to onlookers, *including* all legitimate authorities except AS_h. Note also that the identity of AS_r is not disclosed to an onlooker located between AS_h and AS_r. The basic requirement is that the user's device must store his home domain public key P_h on a long-term basis. We now turn to the details of the protocol:

1. The user begins by generating a nonce N_u and his location-dependent key K_{ur} and storing them in his device. Next, he computes both his alias $P_h(N_u, N_u \bigoplus Uid))$, and his one-way authentication message using his computed key K_{ur} (the nonce used to compute the alias and $AUTH_{ur}$ are different). Then, he sends these messages to the local AS_r along with the identity of AS_h. Note that at this step, the relationship between the user and AS_h is revealed, but we can add another level of privacy as described in Section 7.2.

2. Upon receipt of the initial message, AS_r issues a nonce N_r, and saves $P_r(N_r)$ as well as the future identification of the user in the remote domain, e.g $Uid_r = F(P_h(N_u, N_u \bigoplus Uid))$ in its database (the reasons for these computations will be explained below). Next, it generates its own alias $P_h(N_r, N_r \bigoplus AS_r)$, then computes its authentication message by replacing the timestamp of the token in $AUTH_{rh}$ by $AUTH_{ur}$ in order to prevent a guessing attack on $AUTH_{ur}$. Further details on this token chaining technique are described in [5].

3. When AS_h receives the message from AS_r (flow 2), it proceeds as follows:
   (a) It deciphers the user's alias with S_h to obtain $N_u, N_u \bigoplus Uid$. Then Uid is obtained by applying the xor operation once again.
   (b) AS_h recovers the identity of AS_r in the same way.

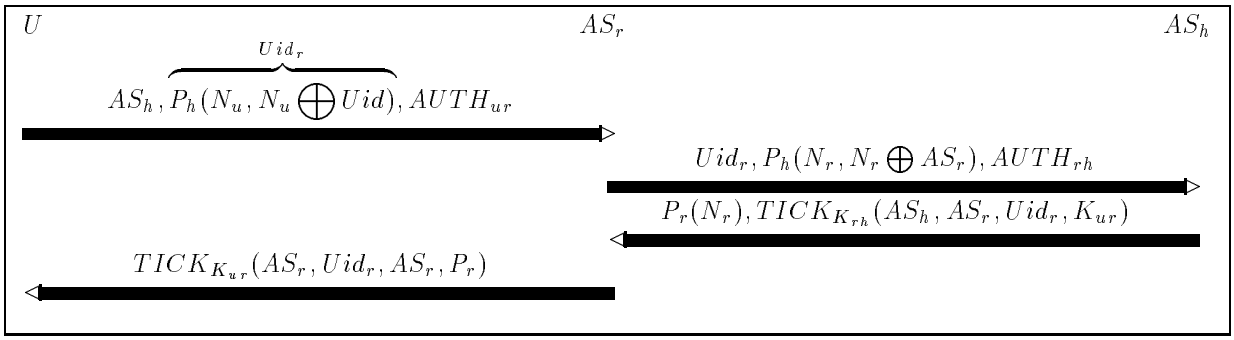

Figure 3: Basic Untraceable Authentication Protocol

(c) Having Uid and AS_r, AS_h is able to look for the corresponding shared secret keys in its database. Next, AS_h generates K_{ur} then recomputes $AUTH_{ur}$ and the chaining token $AUTH_{rh}$.

(d) A match in this last step authenticates both the user and AS_r without revealing either Uid or AS_r to a third-party.

(e) As AS_h needs to send a ticket containing the location dependent key K_{ur} to AS_r, it simply returns N_r enciphered with with AS_r's public key along with the ticket.

4. Upon receiving the message from AS_h (flow 3), AS_r looks for $P_r(N_r)$ in its database and retrieves the necessary information in order to read the incoming ticket. Having K_{ur}, AS_r is able to check the integrity of the key by recomputing $AUTH_{ur}$ received. In fact, sending $P_r(N_r)$ avoids the need for AS_h to compute and send its alias in flow 3. This value can be seen as a secret transaction number which identifies the authentication process involving both the user and AS_h. In other words, it allows AS_r to know who is sending the ticket and to whom K_{ur} belongs while insuring anonymity to the home AS as well as to the user.

   The reason for AS_r to record the encrypted form of N_r is to avoid having to decipher it upon receiving the response from AS_h. This has the advantage of reducing the computation operation with the private key S_r, as an immediate comparison can be done.

5. The fourth flow is purely optional as it allows AS_r to give the user his public key P_r (P_r can be given to the user by another means). This key will be used by the user during future authentications to AS_r for the computation of new aliases.

Once the user has established residence in the remote domain with the identification Uid_r , e.g. $F(P_h(N_u, N_u \bigoplus Uid))$, he may protect this identity on the next single-sign-on using the same alias computation technique as in Section 6.3. The user issues a new nonce N_1 and provides the following message along with his authentication message:

$$P_r(N_1, N_1 \bigoplus Uid_r)$$

Upon receiving this message, AS_r is able to recover Uid_r. Therefore, this alias changing technique allows the user to vary his identity in the remote domain at every authentication process.

7.2 Enhancing the Anonymity Level

The basic protocol does not provide secrecy of the relationship between the user and his home because the identity of AS_h is revealed in flow 1. In order to have an additional degree of privacy as described in Section 3, the user needs to compute an alias for AS_h using AS_r's public key (P_r). As the mobile user does not necessarily have P_r, before the authentication protocol starts the user may obtain AS_r's public key certificate containing P_r either from AS_r or from a public repository.

7.3 A Full Untraceable Authentication Protocol

So far, we have presented two protocols forcing mobile users to contact their home domain for the purpose of authentication. Figure 5 depicts an authentication protocol which avoids having to "call home." The following additional notation is used:

- $\overline{X}$ – Alias computed for identity X using the public key technique as defined in section 6.3.
- Uid_d – Identification/alias of the user in domain D
- $\hat{K}_{ua}$ – A time-dependent key used only once, $\hat{K}_{ua} = F(K_{ua}, T_u, Uid_a)$

The basic idea of this protocol is that the user needs only request a recently visited domain A to guarantee his solvency to the domain currently being visited, B. We assume that the user has already been authenticated in domain A and shares a secret key K_{ua} with AS_a[4]. As the home is no longer involved in the protocol, each domain must generate its own key to be shared with the user while he is visiting its domain.

[4]This step may have involved the home domain using the basic protocol.

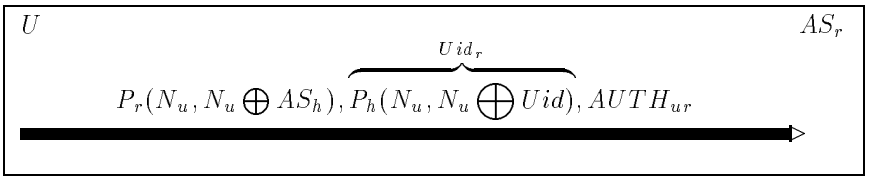

Figure 4: Protocol Hiding Identity of Home Domain Authority from Unauthorized Third Parties

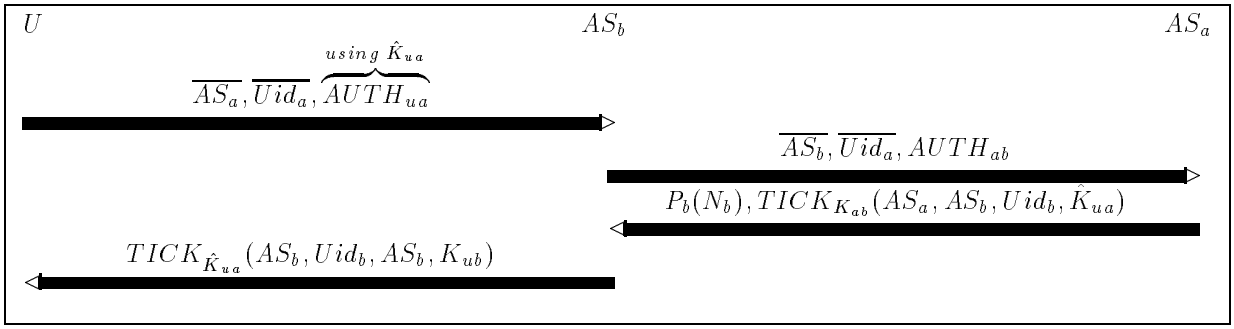

Figure 5: Homeless Authentication Protocol

Therefore, the user provides the last alias used in domain A, along with $AUTH_{ua}$ computed with the one-time key $\hat{K}_{ua}$. This key allows AS_b to give his key K_{ub} to the user in flow 3.

This protocol achieves class C_4 anonymity as the user's migration is hidden from his home domain. One may think that the first foreign authority to execute the protocol will learn the user's home location, as the home authority acts as the trusted third party between between them. However, assuming that the home domain is not in collaboration with the other authorities[5], there is no means for the foreign authority to know that the trusted third party is in fact the home domain. What the foreign authority knows is that the user has provided the identification of a trusted server which can guarantee his solvency.

The class C_5 untraceability is provided partly by this protocol. The home domain will know about the first migration of the user in a remote domain. This is the only information that the home can obtain in a limited time assuming no collusion among domain level authorities. Also, each domain server may know *some* information about the user's movements, namely the domain which provided the authentication to it and the domains to which it was subsequently asked to provide authentication about the user. This protocol is better characterized as between classes C_4 and C_5.

8 Protocol Evaluation

In this section we contrast our proposal with other possible designs. We also evaluate it with respect to our design goals.

[5] AS_h does not notify AS_r of its affiliation with the user.

8.1 Other Possible Designs

Other solutions to the anonymity problem that make use of short-lived aliases are also possible.

One intuitive approach is to have a pre-computed list of aliases kept on the user equipment and his home AS. However, this requires common state to be shared between the mobile equipment and the home domain. The aliases can only hide the identity of the user in the foreign domain, but not his relationship with the home authority.

Another problem arises when all aliases of a list have been used: in this case, the home domain must generate a new alias list and communicate it to the mobile unit. This requires either a secure channel between the user and his home, or an additional secure protocol to transfer the new aliases. These features are not always available in a mobile environment.

A final remark on such solutions is that the mobile equipment and the home AS must be continuously synchronized in order to choose the same alias in the same time. Otherwise, an additional mechanism is needed for the recovering the common state when synchronization is lost. Our alias computation method avoids all these constraints.

In particular, our solution avoids the need for synchronization altogether by using public key cryptosystems. Another approach is to make use of something that is typically already synchronized in distributed system. Herzberg et al [15] propose a method where a user's short-lived alias is computed as a function of the current time (measured with a fairly rough granularity, say to the hour) and the user's secret key (known only to the user and his home AS). The home AS continually recomputes (in the case of our example, every

hour) the mapping between each user and his current alias.

Another technique proposed in the same paper is to compute short-lived aliases by encrypting the user's real identity by a secret key known only to the home AS. During every authentication, the home AS provides the next alias (appropriately concealed from prying eyes) to the user. This approach too avoids the need for synchronization altogether. In particular, it allows the home AS to change its secret key fairly easily unlike in our approach. However, their solutions provide only class C_3 anonymity and cannot be easily upgraded to provide higher degrees of anonymity.

8.2 Computational Complexity of the Protocols

The expensive computations of a public key cryptosystem remain the private key operations. Public key algorithms such as RSA [12] choose the keys in order to minimize both the signature verification process and the public key encryption process. In the case of the low exponent RSA algorithm, the encryption takes only two modular multiplications and thus minimizes the computation on the mobile unit.

In addition to this, new public key algorithms with lower complexity have been developed for wireless networks and are efficient in real time [13]. Therefore, in evaluating the efficiency of the protocol, we count only the total number of the private key operations.

In the basic solution, AS_h needs to decipher both the user's alias and that of AS_r. In the enhanced version of the protocol, one additional operation is needed, as AS_r needs to perform another decryption operation on the alias of AS_h.

Thus, there are two private key operations in the basic solution and three private key operations in the second solution. To put this in perspective, in another protocol based on a combination of the Diffie-Hellman key agreement protocol using digital signatures for authenticity [14], a total of six computationally expensive operations is needed, but such a scheme does not provide anonymity to users.

8.3 Evaluation of the Alias Solution

The alias computation solution meets the design goals as described below:

- **One-time-use alias.** The first time the user accesses the network, he computes an alias. On a subsequent access he is endowed with a new alias by using the chaining alias technique.

- **No direct relationship between aliases.**

  To the extent that the random number N_u in $P_h(N_u, N_u \bigoplus Uid)$ is generated by a good pseudo-random number generator, it is impossible to establish any correlation between the aliases of the same user and to predict the future values of aliases based on its past values.

- **Domain separation.**

  Even in the presence of conspiracy involving every visited AS, it is impossible to determine any relationship between aliases (random numbers) of different domains and to derive the real user's identity from them, provided that the home is not part of the conspiracy[6]. Only the home domain is able to recover the real user's identity from the first alias transmitted by her.

In order to minimize the computation at the mobile unit during call setup, the device can pre-compute a set of aliases when idle. This measure will increase the efficiency of the protocol, but will require additional non-volatile memory in the user equipment.

9 Conclusion

This paper discusses the traceability of mobile users and its implications. It presents a general classification of untraceability requirements and specifically identifies five common classes of untraceability requirements. Existing solutions for preventing the unauthorized tracking of a user's migration in systems such as GSM and CDPD exhibit some weaknesses in providing users good anonymity. We have presented an efficient method for the computation of aliases that hides the identity of every entity involved in the authentication process. The method allows the user to change his alias without being aware of the user's real identity. Therefore, we have provided a new set of untraceable authentication protocols that maintain a strict separation of domains, and avoid sharing domain-specific security information. These authentication protocols are discussed in framework of five classes of untraceability requirements. The proof of the basic protocol in Appendix A does not address the question of privacy. We need an extended logic to reason about the preservation of anonymity. The classification of various levels of anonymity described earlier, and the notation used to describe the various levels, can be a starting point for such extensions. This is a focus of our current work.

References

[1] M. Rahnema, *Overview of the GSM System and Protocol Architecture*, IEEE Communications Magazine, April 1993.

[2] J. Steiner, C. Neuman, J. Schiller, *Kerberos: An Authentication Service for Open Network Systems*, Proceedings of USENIX Winter Conference, February 1988.

[3] R. Molva, G. Tsudik, E. Van Herreweghen, S. Zatti, KryptoKnight *Authentication and Key Distribution System*, Proceedings of ESORICS'92, November 1992.

[4] R. Rivest, *The MD5 Message Digest Algorithm*, Internet DRAFT, July 1991.

[5] R. Molva, D. Samfat, G. Tsudik, *Authentication of Mobile Users*, IEEE Network Magazine, Special Issue on Mobile Communications, March/April 1994.

[6] W. Diffie and M. Hellman, *New Directions in Cryptography*, IEEE Transactions on Information Theory, November 1976.

[6] As noted in Section 7.3, successive domains along a user's path can collude to infer *some* information about the user's movements.

[7] National Bureau of Standards, *Federal Information Processing Standards*, National Bureau of Standards, Publication 46, 1977.

[8] *Cellular Digital Packet Data (CDPD) System Specification*, Release 1.0, July 19, 1993.

[9] European Telecommunications Standards Institute, *Universal Personal Telecommunications*, ETSI NA7 WP1, November 1992.

[10] D. Chaum, A. Fiat and M. Naor, *Untraceable Electronic Cash*, Proceedings of Crypto'88, August 1988.

[11] D. Chaum, *Security Without Identification: Transactions Systems to Make Big Brother Obsolete*, CACM Vol. 28, No. 10, October 1985.

[12] RSA Data Security Inc., *The RC4 Encryption Algorithm*, Document No. 003-013005-100-000-000, March 12, 1992.

[13] M J.Beller, L F. Chang, Y. Yacobi *Security for Personal Communications Services: Public-Key vs. Private Key Approaches* Proceedings of 2nd International Symposium on Personal, Indoor and Mobile Radio Communications, October 1992.

[14] W. Diffie, P.C. van Oorschot, M.J. Wiener *Authentication and Authenticated Key Exchanges in Designs, Codes and Cryptography* Kluwer Academic Publishers, July 1992.

[15] A. Herzberg, H. Krawczyk, G. Tsudik *On Traveling* Incognito Proceedings of First IEEE Workshop on Mobile Computing and its Applications, December 1994.

[16] Michael Burrows et al., *A Logic of Authentication*, Digital Systems Research Center, Technical Report 39, February, 1990, May, 1994.

A Proof of the Basic Authentication Protocol

In proving KryptoKnight-like protocols using something like the BAN logic [16], we have to write new rules for the logic and/or express each protocol step in a form that is more conducive to BAN logic. We have chosen the latter approach.

The first issue is dealing with message authentication codes(MACs). An analogue of the *Message Meaning Rule* for MACs will be as follows: if A gets a message $< X >_K$ from B, where $< X >_K$ is a MAC computed on X using key K, and A believes that it shares the secret key K with B, then A can conclude that B once generated the message X. In other words,

$$\frac{A \textbf{ believes } A \overset{K}{\longleftrightarrow} B,\ A \textbf{ sees } < X >_K}{A \textbf{ believes } B \textbf{ once said } X}$$

In the original BAN logic paper [16], the notation $< X >_K$ is used for "X combined with the formula Y." We use the same notation in a more general sense to include MACs.

The second issue is to decide how to represent the $\bigoplus$ operation. We have decided to use the notation $\{M\}_K$ to also include the following additional case:

- $X,\ E_K(X) \bigoplus M$ i.e: to recover M from the above message, the recipient has to know K. In other words, this is equivalent to "encrypting" X with K.

  In the original description of the BAN logic, $\{M\}_K$ was intended to mean "encryption of M with K." We now extend this to mean "*M protected* by K."

We use the concatenation operator to indicate the situation where both the MAC authentication and the secret protected by a subsequent XOR are relevant. This operator indicates that its components should be considered constitute a single message. The message, $X,\ E_K(X) \bigoplus M$ is represented by $< X >_K \cdot \{M\}_K$.

In notation more suitable to BAN logic, the notation in Section 7.1 can be reduced to the following (the information lost in this reduction is not pertinent to the proof using BAN logic):

- $Token_K(A,\ B,\ C) = < A,\ B,\ C >_K$

  i.e. A token is computed by "protecting" its inputs with the specified key.

- $AUTH_{ab} = N_1,\ N_2, < A,\ N_1,\ N_2 >_{K_{ab}}$

- $Ticket(A,\ B,\ C,\ K_s) = < A,\ C,\ N_a,\ N_b >_{K_{ab}} \cdot \{\ K_s\}_{K_{ab}}$

Armed with these, we can denote the *Basic Untraceable Protocol* as shown in Figure 6 (for simplicity, we use D to mean AS_D).

This protocol is exactly the same protocol as the one described in the paper.

The BAN logic proof proceeds by first writing down the idealized protocol from the real protocol, identifying the initial assumptions and representing them as statements in the logic, using logical inference to derive new statements representing beliefs until the beliefs corresponding to the goals of the protocol are reached.

In this case, our goal is to assure both U and R about the mutual belief in the shared, location-specific, secret key K_{ur}. In formal terms, we need to reach the statements:

U **believes** R **believes** $U \overset{K_{ur}}{\longleftrightarrow} R$,

R **believes** U **believes** $U \overset{K_{ur}}{\longleftrightarrow} R$.

The idealized protocol is described in Figure 7.

The assumptions are as follows.

Keys

A1: U **believes** $U \overset{K_{ur}}{\longleftrightarrow} R$
A2: H **believes** $U \overset{K_{uh}}{\longleftrightarrow} H$
A3: H **believes** $R \overset{K_{rh}}{\longleftrightarrow} H$
A4: R **believes** $R \overset{K_{rh}}{\longleftrightarrow} H$
A5: R **believes** H **controls** $R \overset{K}{\longleftrightarrow} U$
A6: R **believes** $\overset{P_h}{\longmapsto} H$
A7: H **believes** $\overset{P_h}{\longmapsto} H$

Nonces

A8: U **believes** $\overset{P_h}{\longmapsto} H$
A9: U **believes** T_u **is fresh**
A10: H **believes** T_u **is fresh**
A11: R **believes** T_u **is fresh**

M1: $U \Rightarrow R : \overbrace{\{U,\ N'_U\}_{P_h}}^{U's\ alias},\ N_u,\ T_u,\ \overbrace{< U,\ N_u,\ T_u >_{K_{uh}}}^{N'_r}$

M2: $R \Rightarrow H : \overbrace{\{U,\ N'_U\}_{P_h}}^{U's alias},\ \overbrace{\{R,\ N'_R\}_{P_h}}^{R's\ alias},\ T_u,\ N_u,\ N_r,\ N'_r,\ < R,\ N_r,\ N'_r >_{K_{ur}}$

M3: $H \Rightarrow R : \{N_r\}_{P_r},\ < H,\ U,\ N_h,\ N_r >_{K_{rh}} \cdot \{K_{ur}\}_{K_{rh}}$

M4: $R \Rightarrow U : \{R,\ U,\ N_r,\ N_u\}_{K_{ur}} \cdot \{P_r\}_{K_{ur}}$

Figure 6: The Real Protocol

M1: $U \Rightarrow R : < U,\ N_u,\ T_u,\ U \overset{K_{ur}}{\longleftrightarrow} R >_{K_{ur}},\ \{U,\ N'_u\}_{P_h}$

M2: $R \Rightarrow H : \{U,\ N'_u\}_{P_h},\ \{R,\ N'_r\}_{P_h},\ \overbrace{< U,\ N_u,\ T_u,\ U \overset{K_{ur}}{\longleftrightarrow} R >_{K_{ur}}}^{N'_r},\ < R,\ N_r,\ N'_r >_{K_{rh}}$

M3: $H \Rightarrow R : < H,\ U,\ N_h,\ N_r >_{K_{rh}} \cdot \{U \overset{K_{ur}}{\longleftrightarrow} R\}_{K_{rh}}$

M4: $R \Rightarrow U : < R,\ U,\ N_r,\ N_u,\ U \overset{K_{ur}}{\longleftrightarrow} R >_{K_{ur}} \cdot \{\overset{P_r}{\longmapsto} R\}_{K_{ur}}$

Figure 7: The Idealized Protocol

A12: U **believes** N_u **is fresh**
A13: R **believes** N_r **is fresh**

The Proof is as follows:
$M1$

$\Rightarrow$ **R1** : R **sees** $< U,\ N_u,\ T_u,\ U \overset{K_{ur}}{\longleftrightarrow} R >_{K_{ur}}$
$M2$, $A7$, and the *Component Visibility Rule*

$\Rightarrow$ **R2** : H **sees** $U,\ R$
Now, since H knows the identities of the parties involved, it can compute K_{ur} using its definition:

$\Rightarrow$ **R3** : H **believes** $U \overset{K_{ur}}{\longleftrightarrow} R$
$M2$, $R3$, $A10$, the *Message Meaning Rule* and the *Nonce Verification Rule*

$\Rightarrow$ **R3** : H **believes** U **believes** $U \overset{K_{ur}}{\longleftrightarrow} R$
Also, at this point, H can conclude that N'_r **is fresh**.

$\Rightarrow$ **R4** : H **believes** N'_r **is fresh**
$M2$, $A3$, $R4$, the *Message Meaning Rule*, and the *Nonce Verification Rule*

$\Rightarrow$ **R5** : H **believes** R **believes** N_r
$M3$, $A4$, $A13$, the *Message Meaning Rule*, and the *Nonce Verification Rule*

$\Rightarrow$ **R6** : R **believes** H **believes** $U \overset{K_{ur}}{\longleftrightarrow} R$
$R6$, $A5$, and the *Jurisdiction Rule*

$\Rightarrow$ **R7** : R **believes** $U \overset{K_{ur}}{\longleftrightarrow} R$
$R7$, $R1$, $A11$, the *Message Meaning Rule*, and the *Nonce Verification Rule*

$\Rightarrow$ **R7** : R **believes** U **believes** $U \overset{K_{ur}}{\longleftrightarrow} R$
$M4$, $A1$, $A12$, the *Message Meaning Rule*, and the *Nonce Verification Rule*

$\Rightarrow$ **R8** : U **believes** R **believes** $U \overset{K_{ur}}{\longleftrightarrow} R$

Results $R7$ and $R8$ are the goals of the protocol. Using the BAN logic, we have proved that the basic protocol is correct in that U and R authenticate each other if a protocol run is successful.